

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 10-307760

(43)Date of publication of application : 17.11.1998

(51)Int.Cl. G06F 12/16
 G06F 11/10
 G06F 11/22
 G06F 13/00

(21)Application number : 09-119039

(71)Applicant : SONY CORP

(22)Date of filing : 09.05.1997

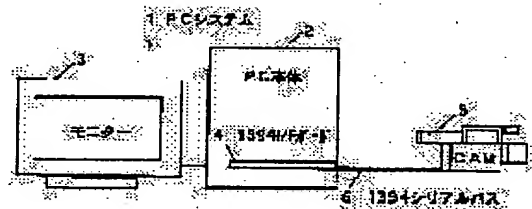
(72)Inventor : IIJIMA YUKO

(54) METHOD AND DEVICE FOR INSPECTION OF CONFIGURATION ROM

(57)Abstract:

PROBLEM TO BE SOLVED: To automatically inspect a configuration ROM in an equipment corresponding to an IEEE1394 high speed serial bus.

SOLUTION: A configuration ROM inspection application in a personal computer body 2 is started to start inspection. An 1394 interface board 4 reads out the whole information based on a directory in a configuration ROM built in a camera incorporated type video tape recorder 5, a leaf offset value or length information written in the ROM through an E394 bus 6. The board 4 executes CRT calculation based on the read information to inspect the reliability of the read data. The whole information is displayed together with text information indicating the meaning of data so that the structure of the configuration ROM can be also visually understood, and when an error exists in the configuration ROM information, the error information also is displayed.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

This Page Blank (uspto)

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平10-307760

(43) 公開日 平成10年(1998)11月17日

(51) Int.Cl.⁹
 G 0 6 F 12/16 3 2 0
 11/10 3 3 0
 11/22 3 5 0
 13/00 3 5 3

F I
 G 0 6 F 12/16 3 2 0 D
 11/10 3 3 0 B
 11/22 3 5 0 B
 13/00 3 5 3 U

審査請求 未請求 請求項の数 8 O L (全 9 頁)

(21) 出願番号 特願平9-119039

(22) 出願日 平成9年(1997)5月9日

(71) 出願人 000002185

ソニー株式会社

東京都品川区北品川6丁目7番35号

(72) 発明者 飯島 祐子

東京都品川区北品川6丁目7番35号 ソニー株式会社内

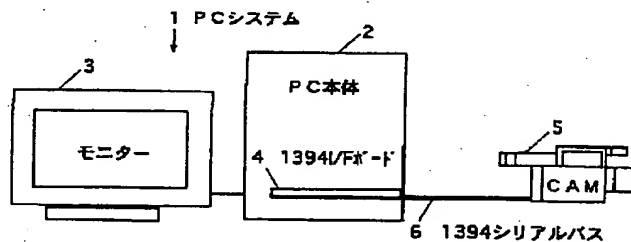
(74) 代理人 弁理士 杉山 猛

(54) 【発明の名称】 コンフィギュレーションROMの検査方法及び装置

(57) 【要約】

【課題】 IEEE 1394 高速シリアルバス対応の機器内のコンフィギュレーションROMの検査を自動的に行う。

【解決手段】 PC本体2内のアプリケーションメモリ上のコンフィギュレーションROM検査用アプリケーションを立ち上げ、検査を開始する。1394インターフェースボード4は、1394バス6を介してCAM5内のコンフィギュレーションROM内のディレクトリーやリーフのオフセット値やそこに書かれている長さ情報等をもとに、全体の情報を読み出す。また、読み出した情報をもとにCRC計算を行い、その読み出したデータの信頼性を検査する。また、さらに視覚的にもコンフィギュレーションROMの構造が理解しやすいように、データの意味を示すテキスト情報と共に全体を表示し、コンフィギュレーションROM情報にエラーがあった場合には、エラー情報も表示する。



【特許請求の範囲】

【請求項1】 IEEE1394高速シリアルバス対応の機器に内蔵されているコンフィギュレーションROMを検査する方法であって、

前記コンフィギュレーションROM内の全ての情報を、前記コンフィギュレーションROMに書かれているディレクトリー、リーフの長さ及びオフセット値等の情報をもとに前記バスを介して読み出すことを特徴とするコンフィギュレーションROMの検査方法。

【請求項2】 前記コンフィギュレーションROM内の情報を全て読み出した後、CRC計算を実行することにより、コンフィギュレーションROM内の情報が正しいかどうかを判断する請求項1に記載のコンフィギュレーションROMの検査方法。

【請求項3】 読み出したコンフィギュレーションROM内の全情報を、オフセット値及び各情報の意味と共に表示し、かつ読み出した情報が正しいかどうかを表示する請求項2に記載のコンフィギュレーションROMの検査方法。

【請求項4】 前記の表示した全情報をファイルにセーブする請求項3に記載のコンフィギュレーションROMの検査方法。

【請求項5】 IEEE1394高速シリアルバス対応の機器に内蔵されているコンフィギュレーションROMを検査する装置であって、

前記コンフィギュレーションROM内の全ての情報を、前記コンフィギュレーションROMに書かれているディレクトリー、リーフの長さ及びオフセット値等の情報をもとに前記バスを介して読み出すことを特徴とするコンフィギュレーションROMの検査装置。

【請求項6】 前記コンフィギュレーションROMから読み出した情報のCRC計算を実行する手段を有する請求項5に記載のコンフィギュレーションROMの検査装置。

【請求項7】 読み出したコンフィギュレーションROM内の全情報を、オフセット値及び各情報の意味と共に表示し、かつ読み出した情報が正しいかどうかを表示する手段を有する請求項6に記載のコンフィギュレーションROMの検査装置。

【請求項8】 前記の表示した全情報をファイルとして記憶する手段を有する請求項7に記載のコンフィギュレーションROMの検査装置。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】 本発明は、IEEE1394高速シリアルバス対応の機器を開発する際に、IEEE1394で規定しているプロトコルを満たしているかどうかを検査するための方法及び装置に関する。

【0002】

【従来の技術】 IEEE1394高速シリアルバス（以

下1394シリアルバスという）を用いてデジタルビデオ信号及びデジタルオーディオ信号の送受信を行う機能を備えたデジタルビデオカメラが既に商品化されている。また、パーソナルコンピュータ（以下PCという）に周辺装置を接続するインターフェースとして1394シリアルバスが注目されている。

【0003】 1394シリアルバス対応の機器内では64ビットのアドレスを使用することができる。そのうち上位16ビットはノードIDを示し、図11に示すようにノードIDに続く48ビットのアドレスの“FFFF F000 0000h”から“FFFF FFFF FFFFh”までのエリアはレジスタ空間と呼ばれており、1394シリアルバスに接続された機器間で共通な情報が書き込まれる。レジスタ空間の“FFFF F000 0400h”からコンフィギュレーションROMが設けられる。

【0004】 図12にコンフィギュレーションROMの構成例を示す。1394シリアルバス対応の機器を開発する際には、コンフィギュレーションROMに正しいデータが書き込まれているかどうかを検査することが必要である。

【0005】 従来は、この検査を以下の手順で実行していた。コンフィギュレーションROMの先頭のオフセット値“FFFF F000 0400h”は決まった値である。また、オフセット値以降に書かれているバス情報ブロック（Bus_Info_Block）やルートディレクトリーの長さ（root_directory_length）等が例えば図12に示したように分かっているならば、そこに書かれている長さ分のデータを読み出せば、コンフィギュレーションROM全体の情報を読み出すことができる。

【0006】

【発明が解決しようとする課題】 しかしながら、従来の方法では、コンフィギュレーションROM全体の情報は簡単に読み出せるが、その情報を人間が見て、ROMの構造やCRCデータが正しいかどうかを判断する必要があった。また、コンフィギュレーションROMに書かれている各種データの長さを判断する必要があった。このため、コンフィギュレーションROMをバージョンアップして内部のデータ構造が変わった場合やデータ構造の分からないコンフィギュレーションROMに対しては正しいかどうかを判断することができなかった。

【0007】 また、長さが合っても、各ディレクトリー（directory）やリーフ（leaf）のオフセットが図12のように連続して取られておらず、飛び飛びになっている場合には、全体の情報を正しく読み出すことができなかった。

【0008】 本発明はこのような問題点に鑑みてなされたものであって、コンフィギュレーションROMに書かれているデータを自動的に読み出すと共に、読み出した

データのチェックやそのチェック結果の表示を自動的に
行えるようにしたコンフィギュレーションROMの検査
方法及び装置を提供することを目的とする。

【0009】

【課題を解決するための手段】本発明では、コンフィ
ギュレーションROM内のディレクトリーやリーフのオフ
セット値やそこに書かれている長さ情報等をもとに、全
体の情報を読み出す。また、読み出した情報をもとにC
RC計算を行い、その読み出したデータの信頼性を検査
する。また、さらに視覚的にもコンフィギュレーション
ROMの構造が理解しやすいように、データの意味を示
すテキスト情報と共に全体を表示し、コンフィギュレ
ーションROM情報にエラーがあった場合には、エラー情
報も表示する。

【0010】

【発明の実施の形態】以下本発明の実施の形態につい
て図面を参照しながら詳細に説明する。

【0011】図1に本発明を適用したシステムの構成を
示す。このシステムはPCシステム1のPC本体2とカ
メラ一体型ビデオテープレコーダ（以下CAMという）
5との間を1394シリアルバス6で接続したものであ
る。そして、PCIバス対応のIEEE1394インター
フェイスボード（以下1394 I/Fボードと略
す）4をPC本体2のPCIスロットに挿入し、PC本
体2上で本発明のアプリケーションを作成し、アシンク
ロナスパケットを送受信することによって、所望の機器
（ここではCAM）のIEEE1394プロトコルを検
査することを実現した例である。なお、この場合のCA
M5が備えているコンフィギュレーションROMに書か
れているデータの構成は図12に示した通りとする。

【0012】図2は図1のPC本体2の内部の構成の概
略を示すものである。ここで、図1と同一の部分には図
1に付した番号と同一の番号が付してある。

【0013】PC本体2の内部に設けられた1394
I/Fボード4には、物理層コントロールブロック（P
HY）11と、リンク層コントロールブロック（LIN
K）12とが設けられている。

【0014】物理層コントロールブロック11は139
4シリアルバスの初期化やバスの使用権の調停等を行
う。また、リンク層コントロールブロック12との間
で、各種制御信号の通信を行うとともに、これらの信号
を1394シリアルバス6に対して送受信する。

【0015】リンク層コントロールブロック12は、パ
ケットの作成/検出、誤り訂正処理等を行う。コンフィ
ギュレーションROMはリンク層コントロールブロック
12の内部に設けられている。

【0016】PCシステム1のPC本体2の内部には、
さらにCPU14と、RAM15と、モニターインター
フェース16と、ファイルメモリ17と、アプリケーシ
ョンメモリ18とが設けられている。

【0017】CPU14はPC本体2の全体の制御等を行
う。RAM15はCPU14が各種データの処理を行
う際のワークエリアとなる。モニターインターフェース
16はモニター3との間の制御信号の通信やモニター3
に対するビデオ信号の送信を行う。ファイルメモリ17
は各種ファイルを格納する。アプリケーションメモリ1
8は各種アプリケーションを格納する。ファイルメモリ
17及びアプリケーションメモリ18は、実際にはハー
ドディスク装置の記憶エリアの一部として構成される。

【0018】RAM15上のアプリケーションプログラ
ムは物理層コントロールブロック11とリンク層コン
trolブロック12の制御、コマンドやレスポンスの作
成等の処理を行う。このアプリケーションプログラムは
コマンドやレスポンスを作成するときにはリンク層コン
trolブロック12内に設けられたレジスタの所定の
アドレスにデータを書き込む。また、他の機器が送信し
たコマンドやレスポンスは、前記レジスタの所定のアド
レスに書き込まれた後、このアプリケーションプログラ
ムにより読み出される。

【0019】なお、実際にはPC本体2内には、さらに
キーボードインターフェースやROM等が設けられてい
るが、ここでは省略した。

【0020】図3は図2に示したPC本体2がコンフィ
ギュレーションROMを検査する際の処理を示すフロー
チャートである。まず、PC本体2内のアプリケーシ
ョンメモリ17上のコンフィギュレーションROM検査用
アプリケーションを立ち上げ、検査を開始する。

【0021】PC本体2内の1394インターフェース
ボード4は、1394バス6を介してCAM5内のコン
フィギュレーションROMの先頭の1クワドレット（q
uadlet）分のデータを読み出し、CRCの長さ
（CRC_Length）とROM_CRC値（rom
_crc_value）をRAM15に保存する（ステ
ップS1）。このCRCの長さには、先頭の1クワドレ
ットを除く、コンフィギュレーションROM全体の長さ
が書かれている。

【0022】コンフィギュレーションROMの先頭の1
クワドレットのデータから、バス情報ブロックの長さ
（Bus_Info_Block_length）が0
4hであると分かるので、さらに4クワドレット分のデ
ータを読み出し、保存する（ステップS2）。

【0023】次に、ルートディレクトリー（root_
directory）の先頭1クワドレット分のデータ
を読み出し、ルートディレクトリーのCRC値を保存す
る（ステップS3）。

【0024】ルートディレクトリーの先頭1クワドレ
ット分のデータから、ルートディレクトリーの長さ（ro
ot_directory_length）が04hで
あると分かるため、さらに4クワドレット分のデータ
を読み出し、保存する（ステップS4）。

【0025】ルートディレクトリー等のディレクトリー(directory)には、さらに他のディレクトリーやリーフへのオフセット値が書かれている可能性があるため、保存しておいたルートディレクトリーのキーデータをもとに、オフセットを指し示すデータがないか検索する(ステップS5)。

【0026】キーデータの構造例を図4に、キータイプ(key_type)のコードを図5に、キータイプ及びキーの値(key_value)のコードを図6に示す。例えばルートディレクトリーの3クワドレット目のキーデータである“8Dh”は、“10001101”である。したがって、先頭の2ビットの“10”(=1)がキータイプのオフセットを示す。そして、残りの6ビットの“001101”(=0D)に対応するエントリーネームはノードユニークIDであり、キータイプがリーフである。つまり、これはノードユニークIDリーフ(Node_Unique_Id_leaf)へのオフセットであることを意味する。同様に、3クワドレット目のキーデータである“D1h”は、ユニットディレクトリー(Unit_Directory)へのオフセットであることを意味する。

【0027】ステップS5での検索の結果、オフセットを指示するデータがあった場合には、そのオフセット値が示すアドレスからデータを読み出し、保存する(ステップS6)。ここでは、ノードユニークIDリーフとユニットディレクトリーを読み出す。この時、ノードユニークIDリーフとユニットディレクトリーのオフセット値を比較して小さいほうのオフセットから読み出す。

【0028】図12では、ノードユニークIDリーフのオフセット値が“05h”であり、ユニットディレクトリーのオフセット値が“01h”であるから、ユニットディレクトリーのオフセット値のほうが小さい。したがって、まずユニットディレクトリーのオフセット値が指し示すアドレスを1クワドレット分だけ読み出すと、ユニットディレクトリーの長さ(Unit_Directory_length)は“02h”であることが分かる。したがって、さらに2クワドレット分のデータを読み出し、保存する(ステップS6)。このとき、ユニットディレクトリーのCRC値も保存しておく。

【0029】ユニットディレクトリーもディレクトリーであるため、ステップS5と同様にしてキーデータをもとに、他のリーフやディレクトリーへのオフセットが書かれていないかチェックする(ステップS7)。

【0030】もし、他のリーフやディレクトリーへのオフセットが含まれていたら、再びまだ読み出していないノードユニークIDリーフのオフセットと新たに検索して発見したオフセットを合わせ、オフセットの昇順に並べ換える。

【0031】図12の場合には、オフセット情報は含まれていなかったため、次にノードユニークIDリーフを

1クワドレット分だけ読み出し、保存する。読み出しによりノードユニークIDリーフの長さ(Node_Unique_Id_leaf_length)は“02h”であることが分かる。したがって、さらに2クワドレット分読み出し、読み出したデータを保存する。このとき、ノードユニークIDリーフのCRC値も保存しておく。

【0032】さらに読み出すべきディレクトリーやリーフのオフセットはこれ以上存在しないため、これでコンフィギュレーションROMの情報を全て読み出すことができた。

【0033】それを確認するために、読み出した全データと保存しておいたCRCの長さ(1加えた値)とを比較する。そして、その値が等しければ良い。もし、この値が等しくなければ、各々のリーフやディレクトリーの長さや情報が間違っているか、あるいはリーフやディレクトリーへのオフセット値が間違っている可能性がある。読み出した長さの正誤情報もチェック結果として、モニターインターフェース15を介してモニター3に表示する(ステップS8)。

【0034】そして、読み出したコンフィギュレーションROMの値をISO/IEC13213で定義されているCRC-16計算式に通して、データの正当性をチェックする。まず、各々のディレクトリー及びリーフ単位でCRC計算を行い、保持しておいた各々のCRC値と等しいかどうかチェックする。全ディレクトリー及びリーフについてのチェックが終わったら、コンフィギュレーションROM全体のCRCを計算し、ROM_CRC値と等しいかどうかチェックする。CRC値の正誤情報もチェック結果として表示する。最後に、読み出したデータにデータの意味を示すテキスト情報も付加して、コンフィギュレーションROMを表にしてモニター3に表示する。その下に読みだしたROM情報のチェック結果も表示する(ステップS8)。

【0035】読み出したデータが正しかった場合のチェック結果の表示例を図7に示す。この表示例では、画面上部に日時等が、中央部にはコンフィギュレーションROM内部のアドレス及びデータ構造等が、下部にはチェック結果が表示されている。そして、チェック結果は、CRC長のチェック結果、バス情報ブロックのROM_CRC値のチェック結果、ルートディレクトリーのCRC値のチェック結果、ユニットディレクトリーのCRC値のチェック結果、及びノードユニークIDリーフのCRC値のチェック結果が全てOKであることが表示されている。

【0036】一方、誤りがあった場合の表示例を図8に示す。ここでは、CRC長のチェック結果とノードユニークIDリーフのCRC値のチェック結果はOKであるが、バス情報ブロックのROM_CRC値のチェック結果及びルートディレクトリーのCRC値のチェック結果

にエラーがあることが表示されている。そして、エラーについてはCRC値について具体的に表示している。

【0037】また、ディレクトリーの階層構造がより深い場合のチェック結果例を正しかった場合について図9及び図10に示す。ここで、図9はモニター3に表示されている画面の上部を示し、図10は下部を示す。

【0038】図7～図10に示したように、チェックした日時やチェック内容のタイトルなども表示しておけば、後でそのファイルを参照したときに、バージョン管理やデバッグ等に非常に有効である。

【0039】以上の検査結果はテキストファイルとしてファイルメモリ16に保存することができる(ステップS9)。

【0040】なお、本実施の形態では、CRCの計算を最後に実行したが、各リーフ/ディレクトリー単位あるいはデータを読み出しながらクワドレット単位で計算してもよい。

【0041】このように、本実施の形態では、コンフィギュレーションROMの各ディレクトリーやリーフ間が連続で確保されていない場合でも、全情報を正確に読み出し、表示することができる。また、キータイプやキーの値をもとに、各ディレクトリー内の情報をテキストでも表示することによって、どのような情報が書かれているのが理解しやすい。

【0042】さらに、全データ及び各ディレクトリー/リーフのCRCチェックを行い、データの信頼性を検査し、結果を表示することによって、エラーの検出も行える。また、検査結果をファイルにセーブすることによって、その場限りの検査ではなく、履歴を残すことができる上に、バグが出た場合にも後でそのデータをもとに原因を追及することが容易にできる。

【0043】さらに、簡単操作かつ短時間でIEEE1394プロトコルを自動的にチェックできる。

【0044】また、IEEE1394コンFORMANCEテスター(Conformance Tester)を操作するユーザーは、プロトコルを知らなくても検査できる。検査結果はテキストファイルとして保存できるため、履歴を残したりデバッグなどに非常に有効に使える。検査用データの文法エラーもチェックできるため、プロトコルを正しく検査でき、かつ、ソースは書き換えなくても、データを書き換えるだけで任意のパケットの送受を行えるため、プロトコル及びセット仕様の変更等

にも柔軟に対応できる。

【0045】

【発明の効果】以上詳細に説明したように、本発明によれば、コンフィギュレーションROMの各ディレクトリー/リーフ構造がコンフィギュレーションROM上に連続的に取られていなくても、正しく全情報を読み出すことができる。また、読み出した結果を、視覚的に理解しやすく表示し、データのチェックまで行ってくれるため、検査装置のユーザーはプロトコルを知らなくても、その仕様を満たしているかどうかをチェックすることができる。

【0046】さらに、その結果をファイルにセーブできるため、デバッグやバージョンの管理などに非常に有効である。

【図面の簡単な説明】

【図1】本発明を適用したシステムの構成を示す図である。

【図2】図1のPC本体の内部の構成の概略を示す図である。

【図3】図2に示したPC本体がコンフィギュレーションROMを検査する際の処理を示すフローチャートである。

【図4】キーデータの構造例を示す図である。

【図5】キータイプのコードを示す図である。

【図6】キータイプ及びキーの値のコードを示す図である。

【図7】読み出したデータが正しかった場合のチェック結果の表示例を示す図である。

【図8】読み出したデータに誤りがあった場合の表示例を示す図である。

【図9】ディレクトリーの階層構造がより深い場合のチェック結果の表示例の画面上部を示す図である。

【図10】ディレクトリーの階層構造がより深い場合のチェック結果の表示例の画面下部を示す図である。

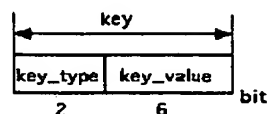
【図11】レジスタ空間のアドレスを示す図である。

【図12】コンフィギュレーションROMの構成例を示す図である。

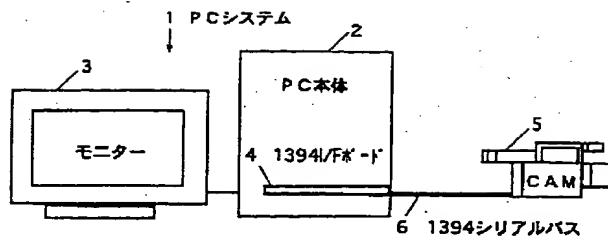
【符号の説明】

1…PCシステム、2…PC本体、3…モニター、4…1394インターフェースボード、6…1394シリアルバス、14…CPU、17…ファイルメモリ、18…アプリケーションメモリ。

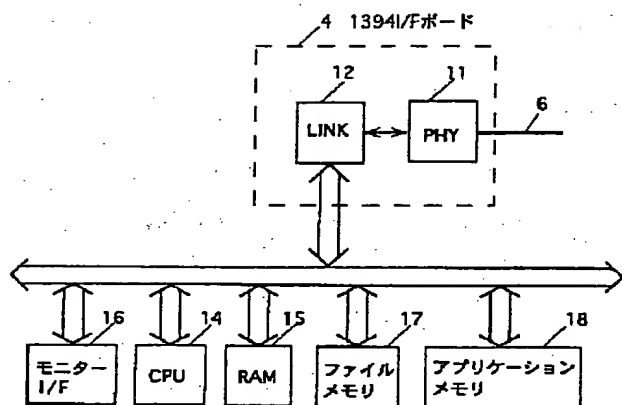
【図4】



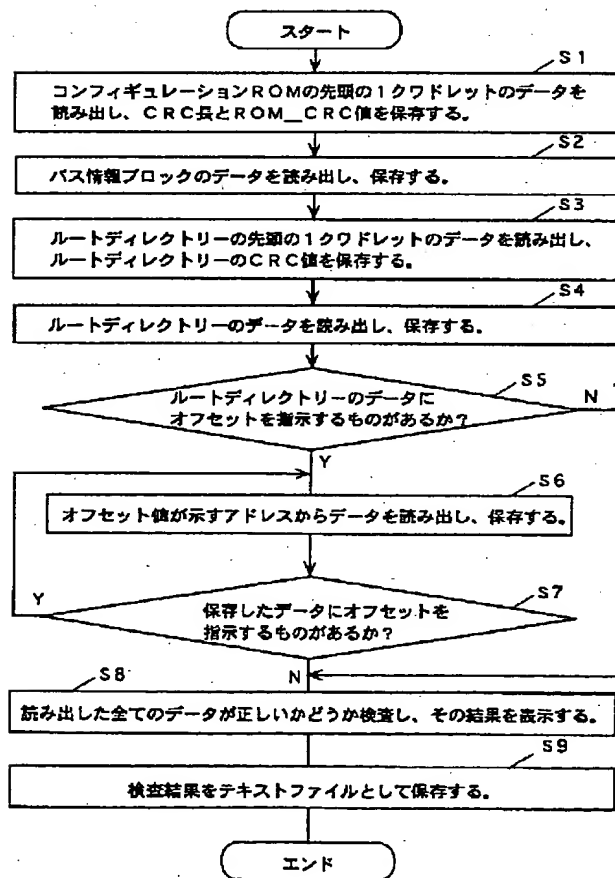
【図1】



【図2】



【図3】



【図5】

Reference Name	Key_type	LSB24ビットの意味
immediate	0	immediate value
offset	1	initial-register-space offset for an immediate value
leaf	2	indirect-space offset for a leaf
directory	3	indirect-space offset for a directory

【図6】

entry name	key_type	key_value
Textual_Descriptor	leaf or directory	01
Bus_Dependent_Info	leaf or directory	02
Module_Vendor_Id	immediate	03
Module_Hw_Version	immediate	04
Module_Spec_Id	immediate	05
Module_Sw_Version	immediate	06
Module_Dependent_Info	leaf or directory	07
Node_Vendor_Id	immediate	08
Node_Hw_Version	immediate	09
Node_Spec_Id	immediate	0A
Node_Sw_Version	immediate	0B
Node_Capabilities	immediate	0C
Node_Unique_Id	leaf	0D
Node_Units_Extent	immediate or offset	0E
Node_Memory_Extent	immediate or offset	0F
Node_Dependent_Info	leaf or directory	10
Unit_Directory	directory	11
Unit_Spec_Id	immediate	12
Unit_Sw_Version	immediate	13
Unit_Dependent_Info	leaf or directory	14
Unit_Location	leaf	15
Unit_Poll_Mask	immediate	16

【図7】

\\\\\\\\1394 Conformance Tester Output File 1996/09/27 08:54:02
[Configuration ROM data of Destination_ID: 0x01]
offset (Rom Base Address FFFF F000 0400h)

0400	Bus_Inf_Blk_len	CRC_length	rom_crc_value	
	04	0F	85	87
0404		1394_Serial_Bus		
	31	33	39	34
0408	rc/cmc/isc/bmc	cyc_clk_acc	max_rec	rsv
	E0	64	30	00
040C		company_id		chip_id_hi
	08	00	46	01
0410		chip_id_lo		
	01	07	01	C9
0414	root_directory_length	CRC_16		
	00	04	FF	7B
0418	Key	Module_Vendor_Id		
	03	08	00	46
041C	Key	Node_Capabilities		
	0C	00	83	C0
0420	Key	Node_Unique_Id_indirect_offset		
	80	00	00	05
0424	Key	Unit_Directory_offset		
	01	00	00	01
0428	Unit_Directory_length	CRC_16		
	00	02	00	9E
042C	Key	Unit_Spec_Id		
	12	00	A0	2D
0430	Key	Unit_Sw_Version		
	13	01	00	01
0434	Node_Unique_Id_leaf_length	CRC_16		
	00	02	DB	0A
0438			46	01
043C			01	C9

<<<< Result >>>>

-- Calculated by ISO/IEC13213 CRC-16 calculation (正式)--

[CRC_length] check OK! (CRC_length=読みだされた長さ)

[CRC check] check OK!

[CRC check] OK! (Root Dir)

[CRC check] OK!

[CRC check] OK!

【図10】

0454	Textual_Descriptor_leaf_length	CRC_16		
	00	03	3A	64
0458		00	00	00
045C		00	00	00
0460		53	4F	59
0464	Bus_Dependent_Info_leaf_length	CRC_16		
	00	06	2D	3A
0468		00	00	00
046C		00	00	00
0470		43	43	4D
0474		44	53	32
0478		30	20	31
047C		30	37	00

<<<< Result >>>>

-- Calculated by ISO/IEC13213 CRC-16 calculation (正式)--

[CRC_length] check OK! (CRC_length=読みだされた長さ)

[CRC check] check OK!

[CRC check] OK! (Root Dir)

[CRC check] OK!

[CRC check] OK!

[CRC check] OK!

[CRC check] OK!

[CRC check] OK!

【図8】

\\\\\\\\1394 Conformance Tester Output File 1996/07/08 12:32:33

[Configuration ROM data of Destination_ID: 0x01]

offset (Rom Base Address FFFF F000 0400h)

0400	Bus_Inf_Blk_len	CRC_length	rom_crc_value	
	04	08	00	29
0404		1394_Serial_Bus		
	31	33	39	34
0408	rc/cmc/isc/bmc	cyc_clk_acc	max_rec	rsv
	E0	64	30	00
040C		company_id		chip_id_hi
	08	00	46	01
0410		chip_id_lo		
	01	06	79	29
0414	root_directory_length	CRC_16		
	00	03	11	80
0418	Key	Module_Vendor_Id		
	03	08	00	46
041C	Key	Node_Capabilities		
	0C	00	83	C0
0420	Key	Node_Unique_Id_indirect_offset		
	80	00	00	05
0434	Node_Unique_Id_leaf_length	CRC_16		
	00	02	3D	65
0438			46	01
043C			06	79

<<<< Result >>>>

[CRC_length] check OK! (CRC_length=読みだされた長さ)

[CRC check] CRC Data ERROR in Configuration ROM!

[CRC check] CRC Data Error! (All of ROM)

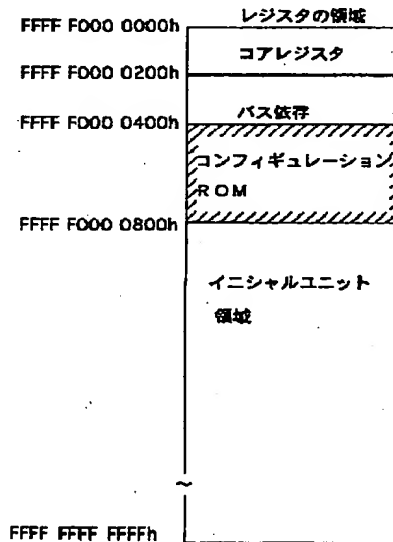
calculated crc_value=76DE (crc_value in ROM=0D29)

[CRC check] CRC Data Error! (Root Dir)

calculated crc_value=D0DE (crc_value in ROM=1180)

[CRC check] OK!

【図11】



【図9】

////// 1394 Conformance Tester Output File 1996/07/08 12:32:33
 [Configuration ROM data of Destination_ID: 0x01]
 offset (Rom Base Address FFFF F000 0400h)

0400	Bus_inf_Blk_len	CRC_length	rom_crc_value	
	04	1E	A2	76
0404		1394_Serial_Bus		
	31	32	39	34
0408	rc/cmc/isc/bmc	cyc_clk_acc	max_rc	rsv reserved
	20	FF	40	00
040C		company_id		chip_id_hi
	08	00	46	02
0410		chip_id_lo		
	00	00	01	50
0414		root_directory_length	CRC_16	
	00	04	C8	0A
0418	Key	Module_Vendor_Id		
	03	08	00	46
041C	Key	Node_Capabilities		
	0C	00	83	CD
0420	Key	Node_Unique_Id_indirect_offset		
	80	00	00	02
0424	Key	Unit_Directory_offset		
	D1	00	00	04
0428		Node_Unique_Id_leaf_length	CRC_16	
	00	02	D9	5C
042C				
	08	00	46	02
0430				
	00	00	01	50
0434		Unit_Directory_length	CRC_16	
	00	03	D7	EE
0438	Key	Unit_Spec_Id		
	12	00	A0	2D
043C	Key	Unit_Sw_Version		
	13	00	01	00
0440	Key	Unit_Dependent_Info_offset		
	D4	00	00	01
0444		Unit_Dependent_Info_dir_length	CRC_16	
	00	03	4E	EA
0448	Key	???	00	00
	40	3C		
044C	Key	Textual_Descriptor_offset		
	81	00	00	02
0450	Key	Bus_Dependent_Info_offset		
	82	00	00	05

【図12】

offset (Base Address FFFF F000 0000h)

04 00h	Bus_Info_Block _length (04h)	CRC_length (0Fh) * 1	rom_crc_value (F7h)	(22h)	
04 04h	"1" (31h)	"3" (33h)	"9" (39h)	"4" (34h)	Bus_Info_Block
04 08h	* 2 (0000)	reserv (0000)	cyc_clk_acc (64h(100))	max _rec (0011)	reserved (00h)
04 0Ch	(08h)	company_id (00h)	(46h)	chip_id_hi (01h)	Bus Info Block _length
04 10h	(01h)	chip_id_lo (00h)	(00h)	(00h)	
04 14h	root_directory_length (00h)	(04h)	(11h)	CRC_16 (80h)	root_directory
04 18h	key (03h)	Module Vendor ID(company_id) (08h)	(00h)	(46h)	root_directory _length
04 1Ch	key (0Ch)	Node Capabilities (00h)	(83h)	(C0h)	
04 20h	key (8Dh)	Node_Unique_Id_indirect_offset (00h)	(00h)	(05h)	
04 24h	key (D1h)	Unit_Directory_offset(Quadlet単位) (00h)	(00h)	(01h)	
04 28h	unit_directory_length (00h)	(02h)	(FAh)	CRC_16 (D0h)	unit_directory
04 2Ch	key (12h)	Unit_Spec_ID(company_id) (00h)	(A0h)	(2Dh)	unit_directory length
04 30h	key (13h)	Unit_Sw_Version (01h)	(00h)	(01h)	
04 34h	node_unique_id_leaf_length (00h)	(02h)	(0Ch)	CRC_16 (23h)	node_unique_id _leaf
04 38h	(08h)	company_id (00h)	(46h)	chip_id_hi (01h)	node_unique_id leaf_length
04 3Ch	(01h)	chip_id_lo (00h)	(00h)	(00h)	

* 1 : 先頭quadletを除くコンフィギュレーションROM全体の長さ。

* 2 :

rc (1h)	cmc (1h)	isc (1h)	bmc (0h)
------------	-------------	-------------	-------------

rc : resolver capable
cmc : cycle master capable
isc : isochronous capable
bmc : bus manager capable

This Page Blank (uspto)